Lecture # 26

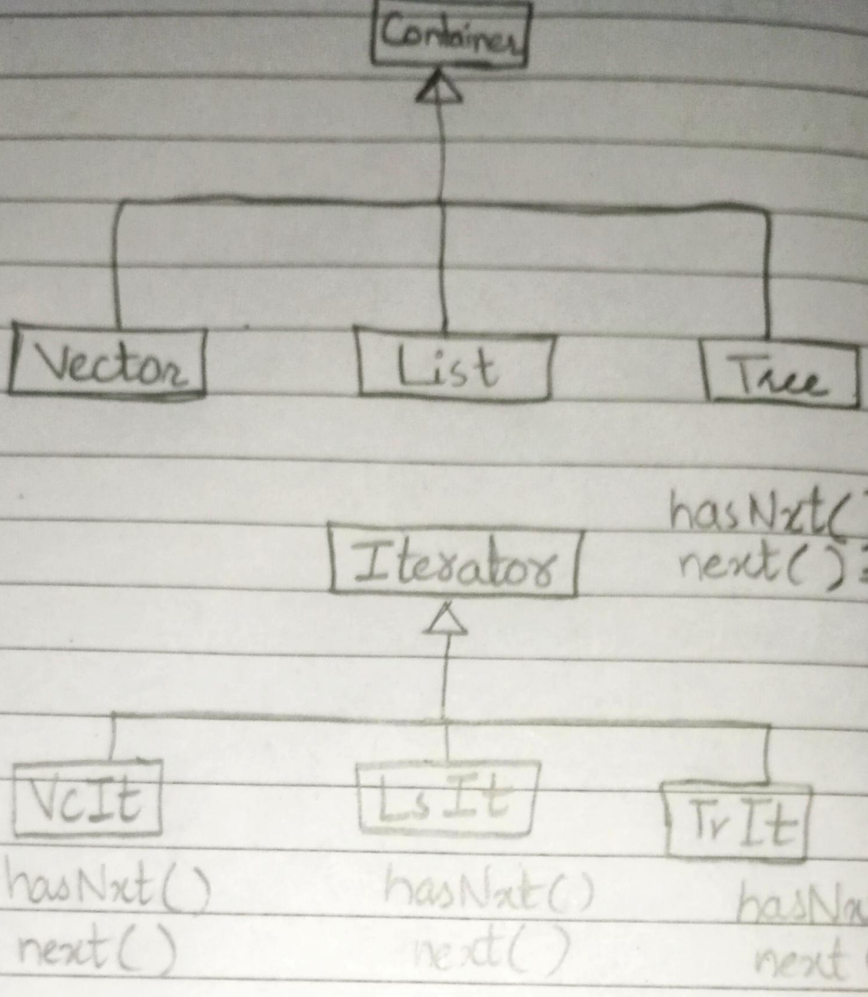
Date: NOV 26, 25 Day: Wednesday

Iterator Design Pattern

```
int sum (node * head){
    int s = 0;
    node * ptr = head;
    while (ptr != NULL){
        s = s + (ptr -> data);
        ptr = ptr -> next;
    }
```

```
int sum (int a[], int n){
    int s = 0;
    for(int i = 0; i < n; i++)
        s = s + a[i];
    return s;
}
```

```
// Solution
int sum (Iterator * it){
    int s = 0;
    while(it -> hasNxt())
        s = s + (it -> next());
    return s;
}
```




```
class LsIt : public Iter {
        node * ptr;
    LsIt (List *ls) {
        ptr = ls -> head;
    }
    bool hasNxt () {
        return (ptr != NULL);
    }
int next () {
    int x = ptr -> data;
    ptr = ptr -> next;
    return x;
} };
```

Lecture #27

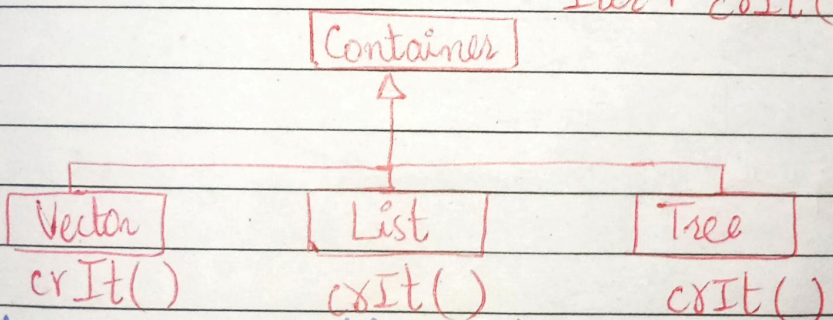
Date: Dec 1st, 25

Day: Monday

Make a vector iterator (iterator for vector class)

```
class VcIt :: public Iter {
        Vector * vec ;
          int i ;
    VcIt ( Vector * v ) {
            vec = v ;
            i = 0 ;
          }

  int next ( ) {
          int temp = i ;
           ++i ;
        return( vec → a [ temp ]) ;
       }
 bool hasNext ( ) {
      return ( i < vec → no ) ;
         }
```

Iter * crIt ( ) = 0




```
class vector :: public container {
      Item * crIt ( )
        return new VcIt ( this ) ;
      }
       } ;
```

Date: Day:

```
class List :: public Cont {
        ---------
        Item * crIt(){
          return new LsIt(this);
        }
};
```

modified

```
int sum (Cont * con){
    Iter * it = con → crIt();
        int s = 0;
    while ( it → hasNext())
        s = s + (it → next());

    return s;
}
```

Binary Tree Iterator :-

```
class BTI :: public Iter {
        Queue < Node * > que;
    BTI(BnTr * tree) {
// enqueue      que.add (tree → root);
    }
                // Level order Traversal
int next (){
    node * curr = que.rem();
    if ( curr → left)
        que.add (curr → left);
```

Date: Day:

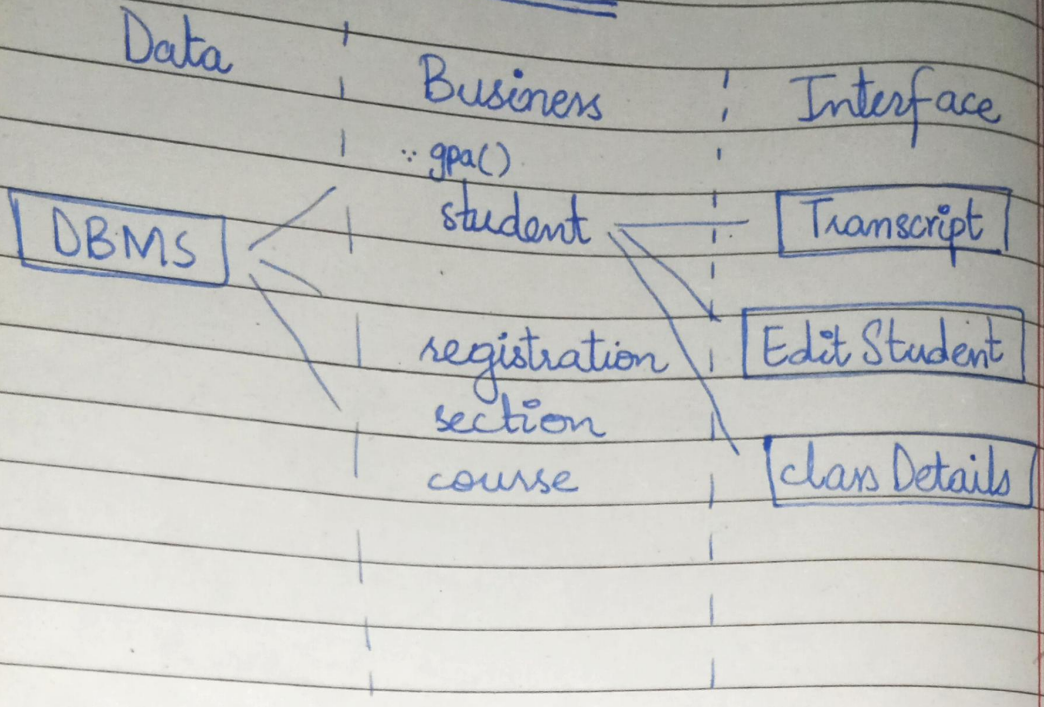
```
if (curr → right)
    que.add(curr → right);

    return curr → data;
    }

bool hasNext() {
    return (!que.isEmpty());
    }
```

Date: Dec 3rd, 2025

Lecture #28 (Last)

Day: Wednesday

Three Tier Architecture



Draw a class diagram for CRS:

→ Old CRS +

→ Semesters +

→ teachers, attendance, Marks, evaluation

→ weights,

→ compute total marks of a stud.

// grade acc. to a given grading Scheme